I Didn't Know You Could Do That! (2023 Edition) "IDKYCDT"

Presenting Tips &Tricks Since 2014

2023 UNITE Conference
Session 6091

Thursday, 9 March 2023, 1:30 p.m.

Topics

- Date Manipulations in WFL
  - Paul Kimpel
- Remote Access to Your Home MCP System
  - Doug Dobson, GoldEye Software

IDKYCDT: Date Manipulations in WFL

Paul Kimpel

2023 UNITE Conference
Session 6091

Thursday, 9 March 2023, 1:30 p.m.

The Genesis

- Don Gregory's virtual file IOH "perverse ideas"
  - UNITE 2002 Conference, Baltimore
  - Suggested using an IOH for special functions in WFL
  - Such as *"How many days is it until Friday?"*
- But that's a nail & sledgehammer approach
  - WFL has INCLUDE files and subroutines
  - Has everything needed to do date calculations
  - *But how?*
- Easy – Algorithm 199
  - Devised by Robert G. Tantzen of Holloman AFB, NM
  - Published in *Comm. ACM*, v6.8, August 1963
  - In ALGOL!

What is Algorithm 199?

- Specifies two procedures
- **Jday**
  - Convert Gregorian (y/m/d) date to Julian Day Number
  - Valid for years 0001 through at least 9999
  - Epoch: Julian 0 => 1 January 4713 BCE (y = -4712)
- **Jdate**
  - Convert Julian Day Number to Gregorian date
- Also specifies “simplified” procedures
  - **Kday**, **Kdate**
  - Work for ‘00-03-01 through ‘99-12-31 of any century
  - Also correct for 1900-03-01 through 2100-02-28 (day numbers 1 – 73049)

Why Are These Useful?

- Convert between a *date* and a *day number*
- Once you have a day number, lots of date manipulations become trivial:
  - Advancing a date ± some number of days
  - Number of days between two dates
  - Day of the week – a simple MOD 7 calculation
  - Handling Ordinal ("Julian") dates – YYDDD
  - What is the last date in a month?
  - When is U.S. Thanksgiving in any given year?
  - What is the Monday for a U.S. Federal Holiday?
  - And… *How many days is it until Friday?*

My Solution

- Separate library files for
  - Kday/Kdate (works within a century)
  - Jday/Jdate (works for the Common Era – A.D.)
  - The K-routines are fine for most applications
- Basic routines
  - **DATEKDAY (GREG VALUE, JULDAY)**
  - **DATEKDATE (JULDAY VALUE, GREG)**
  - For the J-routines – **DATEJDAY/DATEJDATE**
  - Dates are binary integers as YYYYMMDD
- You probably won't need to use these directly

Additional Library Date Routines

- **DATEINCREMENT (GREG1, DAYS, GREG2)**
- **DATEDIFFERENCE (GREG1, GREG2, DAYS)**
- **DATEWEEKDAY (GREG, DOWNR)**
- **DATEGREGORIANTOJULIAN (GREG, YYYYDDD)**
- **DATEJULIANTOGREGORIAN (YYYYDDD, GREG)**
- **DATEVALIDATE (GREG, ISVALID)**
- From these you can build additional higher-level routines

Defined Library Constants

```
DATEMAXV=              21000228 / 99991231
DATEMINV=              19000301 / 00010301
DATEYEARBIASV=         1900

DATESUNDAYV=           0
DATEMONDAYV=           1
DATETUESDAYV=          2
DATEWEDNESDAYV=        3
DATETHURSDAYV=         4
DATEFRIDAYV=           5
DATESATURDAYV=         6
```

Using the Library

```
$$ INCLUDE (LIB)WFL/UTIL/DATE/JMODULE ON PROD
INTEGER TODAY, DOW, JULDAY; STRING DAYNAME;
SUBROUTINE DOWNAME(INTEGER DOW VALUE, STRING DAYNAME);
  BEGIN
  CASE DOW OF
    BEGIN
    (DATESUNDAYV):    DAYNAME:= "Sunday";
    (DATEMONDAYV):    DAYNAME:= "Monday";
    (DATETUESDAYV):   DAYNAME:= "Tuesday";
    (DATEWEDNESDAYV): DAYNAME:= "Wednesday";
    (DATETHURSDAYV):  DAYNAME:= "Thursday";
    (DATEFRIDAYV):    DAYNAME:= "Friday";
    (DATESATURDAYV):  DAYNAME:= "Saturday";
    ELSE:             DAYNAME:= "*unknown*";
    END;
  END DOWNAME;

  TODAY:= DECIMAL(TIMEDATE(YYYYMMDD));
  DATEWEEKDAY(TODAY, DOW);
  DATEJDAY(TODAY, JULDAY);
  DOWNAME(DOW, DAYNAME);
  DISPLAY "Today is " & DAYNAME & ", " & STRING(TODAY,*);
  DISPLAY "The Julian day number is " & STRING(JULDAY,*);

DISPLAY:Today is Saturday, 20230304.
DISPLAY:The Julian day number is 2460008.
```

Answering Don's Friday Question

```
    INTEGER TODAY, DOW, DIFF, NEWGREG;

    TODAY:= DECIMAL(TIMEDATE(YYYYMMDD));
    DATEWEEKDAY(TODAY, DOW);
    DIFF:= (DATEFRIDAYV - DOW + 7) MOD 7;
    DATEINCREMENT(TODAY, DIFF, NEWGREG);
    DISPLAY "It is " & STRING(DIFF,*) & " days until the next Friday, " &
            STRING(NEWGREG,*);
```

```
DISPLAY:It is 6 days until the next Friday, 20230310.
```

Ordinal (“Julian”) Dates

```
    INTEGER TODAY, JULDAY;

    TODAY:= DECIMAL(TIMEDATE(YYYYMMDD));
    DATEGREGORIANTOJULIAN(TODAY, JULDAY);
    DISPLAY "The Ordinal Julian date is: " & STRING(JULDAY,*);

DISPLAY:The Ordinal Julian date is: 2023063.
```

```
    INTEGER RESULT, CDATE;
    FILE F (TITLE=*SYSTEM/WFLSUPPORT ON DISK);

    RESULT:= F(AVAILABLE);
    IF RESULT = 1 THEN
      BEGIN
      DATEJULIANTOGREGORIAN(F(CREATIONDATE), CDATE);
      DISPLAY "File Creation Date = " & STRING(CDATE,*);
      END
    ELSE
      DISPLAY "File not available (" & STRING(RESULT,*) & ")";
    RELEASE(F);

DISPLAY:File Creation Date = 20130218.
```

Computing U.S. Thanksgiving

```
SUBROUTINE THANKSGIVING(INTEGER YEAR VALUE);
  BEGIN
  INTEGER DIFF, DOW, TDAY;
  STRING MSG;

  TDAY:= YEAR*10000 + 1101;                  % FIND THE FIRST THURSDAY
  DATEWEEKDAY(TDAY, DOW);
  DIFF:= (DATETHURSDAYV - DOW + 7) MOD 7;
  DATEINCREMENT(TDAY, DIFF+21, TDAY);        % THEN SKIP 3 WEEKS FORWARD
  MSG:= "Thanksgiving in the year " & STRING(TDAY DIV 10000,*);
  IF TDAY < TODAY THEN
    MSG:= MSG & " was "
  ELSE
    MSG:= MSG & " will be ";

  MSG:= MSG & "November " & STRING(TDAY MOD 100,*);
  DISPLAY MSG;
  END THANKSGIVING;

THANKSGIVING(1789);
THANKSGIVING(2022);
THANKSGIVING(2023);
```

```
DISPLAY:Thanksgiving in the year 1789 was November 26.
DISPLAY:Thanksgiving in the year 2022 was November 24.
DISPLAY:Thanksgiving in the year 2023 will be November 23.
```

Gotchas

- **DATEJDAY** & **DATEJDATE** require 8-digit dates
  - Be careful of dates with 2-digit years (e.g., YYDDD)
- Dates before 0001-01-01 (J=1721426) are not supported
- Gregorian calendar started 15 October 1582
  - Prior were in the Julian calendar
  - A199 computes those as “proleptic” Gregorian dates
    - Calculated as if Gregorian leap-year rules applied
- Invalid date parts (e.g., 2023-02-31)
  - May produce invalid results
  - Use the library’s **DATEVALIDATEDATE** routine

Resources

- Original Algorithm 199 publication (p.444)
  - https://dl.acm.org/doi/10.1145/366707.390020
- Gregory *Journal* article from January 2003
  - https://www.digm.com/Resources/Gregory/Date-Manipulations-in-WFL.pdf
- The Code
  - https://www.digm.com/Resources/Date/

END